Offline - AI Assignment for Candidates

Table of Contents

Document Details

Date	Jul 23, 2024
Author(s)	@Abhinav Tripathi
Revision	V 1.0
Status	RELEASED

Objective

Develop a web application that allows users to upload documents and receive summarized versions using a locally deployed Language Model.

Components:

1. Backend: Python (Flask or FastAPI)
2. Frontend: React
3. LLM Deployment: Locally hosted LLM for document summarization

---

Assignment Details

1

Backend (Python)

- Use Flask or FastAPI to create a backend service.
- Implement an endpoint /upload to handle file uploads (PDF, DOCX, TXT).
- Implement an endpoint /summarize to process and summarize the document using a locally hosted LLM.
- Ensure the backend can handle multiple concurrent requests.

Tasks

- Set up a basic Flask or FastAPI application.
- Implement file handling and storage.
- Integrate a pre-trained LLM (e.g., GPT-2 or smaller model suitable for local deployment).
- Implement a summarization endpoint that returns a summary of the uploaded document.

Evaluation Criteria:

- Code quality and structure.
- Correct implementation of endpoints.
- Handling of file uploads and errors.
- Efficiency and performance of the summarization process.

---

2 Frontend (React)

- Create a simple React application that interacts with the backend.
- Implement a file upload component.
- Display the uploaded file's content.
- Display the summarized text returned by the backend.

Tasks:

- Set up a React application using Create React App or a similar tool.
- Create a form for file uploads.
- Implement API calls to the backend for file upload and summarization.
- Display the summarized text in a user-friendly manner.

Evaluation Criteria:

- Code quality and structure.
- User interface design and responsiveness.
- Correct implementation of API calls.
- Error handling and user feedback.

---

3 LLM Deployment

- Deploy a pre-trained LLM (e.g., GPT-2 or a smaller, local model) locally.
- Ensure the model can process the uploaded documents and generate summaries.
- Optimize the model deployment for efficiency and performance.

Tasks

- Set up a local environment for the LLM (use Docker if possible).
- Load the pre-trained model and ensure it can generate text summaries.
- Integrate the model with the backend to handle summarization requests.

Evaluation Criteria

- Correct deployment and integration of the LLM.
- Performance and efficiency of the model.
- Quality of the generated summaries.

Submission Requirements

- 1 A GitHub Monorepo repository containing:

    - The backend code.
    - The frontend code.
    - Instructions for setting up and running the application locally.
    - A suitable Docker Compose file for local deployment will be a plus.
- 2 A brief document explaining the approach, challenges faced, and how they were overcome, part of the above repo, in markdown.
- 3 A video (optional) demonstrating the working application.
- 4 A Bibliography of all relevant sources (FOSS) forked / referred to.

Deadline

A week from reception of this Problem over e-mail.

END OF DOCUMENT
